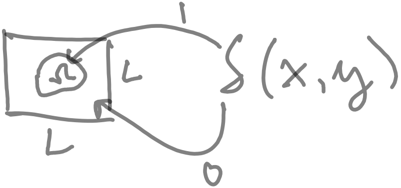
# Computing General Integrals using Monte Carlo

Lets consider integrating $f(x,y)$ on domain $\Omega \subset [0,L]^2$

$$G = \int\int_{\Omega} f(x,y)dxdy = \int_0^L \int_0^L f(x,y)\delta(x,y)dxdy,$$

where $\delta(x,y) = 1$ if $(x,y) \in \Omega$ and $\delta(x,y) = 0$ if $(x,y) \notin \Omega$.

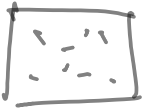



$$p(x,y) = \frac{1}{|\Omega|}\delta(x,y)$$

$Z$ distributed by $p$

$$G = |\Omega| \, E[f(Z)] = |\Omega| \int_0^L \int_0^L f(x,y)\, p(x,y)\, dxdy$$

$$\tilde{p}(x,y) = \frac{1}{L^2}$$



$$\tilde{X}$$

$$G = |\Omega| \int_0^L \int_0^L f(x,y) \frac{p(x,y)}{\tilde{p}(x,y)} \tilde{p}(x,y) \, dx \, dy$$

$$G = |\Omega| E[f(Z)] = |\Omega| E\left[f(\tilde{X}) \frac{p(\tilde{X})}{\tilde{p}(\tilde{X})}\right]$$

$$= |\Omega| E\left[f(\tilde{X}) \frac{p(\tilde{X})}{1/L^2}\right] = |\Omega| L^2 E[f(\tilde{X}) p(\tilde{X})]$$

$$G = |\Omega| L^2 E[f(\tilde{x}) p(\tilde{x})]$$

$$\approx \frac{|\Omega| L^2}{N} \sum_{i=1}^{N} f(s_i) p(s_i)$$

$$= \frac{L^2}{N} \sum_{i=1}^{N} f(s_i) S(s_i)$$

# Monte Carlo Methods: The Good and the Bad

What are some *advantages of MC methods?*

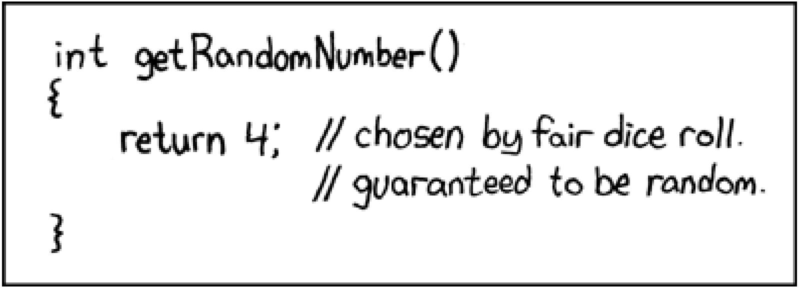
easy to use for integration
reliable convergence

What are some *disadvantages* of MC methods?

slow! especially for simple integration

unreliable due to randomness

# Computers and Random Numbers




[from xkcd]

How can a computer make random numbers?

use an actual source of randomness

# Random Numbers: What do we want?

What properties can 'random numbers' have?

- Have a specific distribution
  (e.g. 'uniform'–each value in given interval is equally likely)
- Real-valued/integer-valued
- Repeatable (i.e. you may *ask* to exactly reproduce a sequence)
- Unpredictable
  - V1: 'I have no idea what it's going to do next.'
  - V2: No amount of engineering effort can get me the next number.
- Uncorrelated with later parts of the sequence
  (Weaker: Doesn't repeat after a short time)
- Usable on parallel computers

# What's a Pseudorandom Number?

Actual randomness seems like a lot of work. How about 'pseudo-random numbers?'

**Idea:** Maintain some 'state'. Every time someone asks for a number:

$$\text{random\_number}, \text{new\_state} \;=\; f(\mathrm{state})$$

Satisfy:

- Distribution
- 'I have no idea what it's going to do next.'
- Repeatable (just save the state)
- Typically *not* easy to use on parallel computers

**Demo:** Playing around with Random Number Generators

# Some Pseudorandom Number Generators

Lots of variants of this idea:

- LC: 'Linear congruential' generators
- MT: 'Mersenne twister'
- almost all randonumber generators you're likely to find are based on these–Python's `random` module, `numpy.random`, C's `rand()`, C's `rand48()`.

# Counter-Based Random Number Generation (CBRNG)

## What's a CBRNG?

**Idea:** Cryptography has *way* stronger requirements than RNGs.
*And* the output *must* 'look random'.

(Advanced Encryption Standard) AES algorithm:
$$128 \text{ encrypted bits} = \mathrm{AES}\,(128\text{-bit plaintext}, 128 \text{ bit key})$$

We can treat the encrypted bits as random:
$$128 \text{ random bits} = \mathrm{AES}\,(128\text{-bit counter}, \text{arbitrary } 128 \text{ bit key})$$

- Just use $1, 2, 3, 4, 5, \ldots.$ as the counter.
- *No* quality requirements on counter or key to obtain high-quality random numbers
- *Very* easy to use on parallel computers
- Often accelerated by hardware, faster than the competition

**Demo:** Counter-Based Random Number Generation

# Outline

# Error in Numerical Methods

Every result we compute in Numerical Methods is inaccurate. What is our model of that error?

computed = real value + error

Suppose the true answer to a given problem is $x_0$, and the computed answer is $\tilde{x}$. What is the absolute error?

absolute error $= |x_0 - \tilde{x}|$

$= |\tilde{x} - x_0|$

# Relative Error

What is the relative error?

relative error $= \dfrac{|\tilde{x} - x_0|}{|x_0|}$

Why introduce relative error?

error $= 2$, $\quad x_0 = 4$

$x_0 = 4{,}000{,}000$

What is meant by 'the result has 5 accurate digits'?

$\tilde{x} = .00005$

$\tilde{x} = x_0 . x_1 x_2 x_3 x_4 \times 10^y$

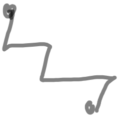
$x_0 \neq 0 \qquad 12345 \ldots\ldots$

# Measuring Error

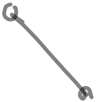
Why is $|\tilde{x}| - |x_0|$ a bad measure of the error?

$$|\tilde{x} - x_0| \neq |\tilde{x}| - |x_0|$$

If $\widetilde{\boldsymbol{x}}$ and $\boldsymbol{x}_0$ are vectors, how do we measure the error?



$$\frac{\|\tilde{x} - x_0\|}{\|\tilde{x}\|} \qquad 2,\ F,\ \infty$$



## Sources of Error

What are the main sources of error in numerical computation?

experimental (input)

truncation error

roundoff error

method error

# Digits and Rounding

Establish a relationship between '*accurate digits*' and rounding error.

$$\tilde{x} = 0.0034$$

$$x_0 = 0.0034271$$

$$|\tilde{x} - x_0| = .0000271\ldots \leq 10^{-4}$$

$$\frac{|\tilde{x} - x_0|}{|\tilde{x}|} \leq 10^{-2}$$

absolute

$$|\tilde{x} - x_0| \leq 10^{r-k}$$

if $\tilde{x}$ has $k$ accurate digits

and $x_0 = X \cdot 10^r$

where $X = 0.\cdots\cdots$

$\tilde{x} = 123450.000$ $\quad k = 5$

$x_0 = 123456.789$ $\quad r = 6$

$|\tilde{x} - x_0| \leq 10$

relative error

$$\frac{|\hat{x} - x|}{|x_0|} \leq \frac{10^{r-k}}{|x_0|} \rightarrow 10^{-k}$$

$$|x_0| \leq 10^r$$

# Condition Numbers

Methods $f$ take input $x$ and produce output $y = f(x)$.
Input has (relative) error $|\Delta x| / |x|$.
Output has (relative) error $|\Delta y| / |y|$.
**Q:** Did the method make the relative error bigger? If so, by how much?

$$\kappa = \frac{\text{relative error of output}}{\text{relative error of input}}$$

$$\kappa = \frac{\dfrac{|f(x+\Delta x) - f(x)|}{|f(x)|}}{|((x+\Delta x)-x)| / |x|}$$

$$r = \frac{\dfrac{|f(x+\Delta x) - f(x)|}{|f(x)|}}{\dfrac{|(x+\Delta x) - x|}{|x|}}$$

$$r = \frac{|x|\,|f(x+\Delta x) - f(x)|}{|f(x)|\,|\Delta x|}$$

$$\kappa \approx \frac{|x|\,|f'(x)|}{|f(x)|}$$

$$f'(x) = \lim_{\Delta x \to 0} \left( \frac{f(x+\Delta x) - f(x)}{\Delta x} \right)$$

# $n$th-Order Accuracy

Often, *truncation error* is controlled by a parameter $h$.

Examples:

- distance from expansion center in Taylor expansions
- length of the interval in interpolation

A numerical method is called '$n$th-order accurate' if its truncation error $E(h)$ obeys

$$E(h) = O(h^n).$$

# Outline